University of Rochester. ECE 449 Machine Vision.

Tiecheng Su, Dept. of Electrical Engineering

HW5. Submission: 18 April 2016

%% Initialization

close all; clc; clear

%%

v = VideoReader('E:/2016Spring/MV/HW5/test5.mp4');

%implay('E:/2016Spring/MV/HW5/test3.mp4');

nframes = get(v, 'NumberOfFrames');

% Use color information to track a uniquely colored foreground blob

close all; clc

singleFrame = read(v,1);

shapeInserter = vision.ShapeInserter('BorderColor','Custom','CustomBorderColor',uint8([255 0 0]));

shapeInserter2 = vision.ShapeInserter('Shape','Circles','BorderColor','Custom','CustomBorderColor',uint8([0 0 255]));

tag = zeros([size(singleFrame) nframes],'uint8');

mx = [];

my = [];

for k = 1:nframes

singleFrame = read(v,k);

%[row,col] = find(singleFrame(:,:,3) <= 11 & singleFrame(:,:,1)>=83);

[row,col] = find(singleFrame(:,:,3) <= 33 & singleFrame(:,:,1)>=114);

rectangle = int32([min(col),min(row),max(col)-min(col),max(row)-min(row)]);

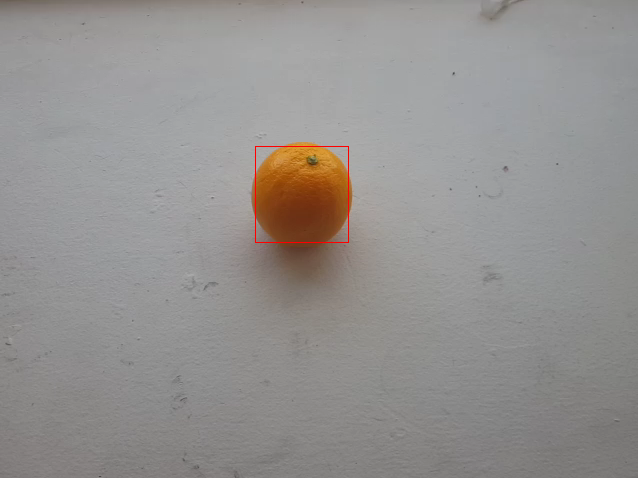
tag(:,:,:,k) = step(shapeInserter,singleFrame,rectangle);

X = [mean(col);mean(row);0;0];

mx = [mx;X(1)];

my = [my;X(2)];

end



% figure

% imshow(singleFrame);

% Implement a 2D Kalman filter tracker

X = [mx(1);my(1);0;0];

P = eye(4);%covariance matrix

dt = 1;

I = eye(4);

A = [1 0 dt 0; 0 1 0 dt; 0 0 1 0; 0 0 0 1];%transition matrix

H = [1 0 0 0; 0 1 0 0];%ouput transition matrix

sigma\_x = 0.001;

sigma\_y = 0.001;

R = [sigma\_x 0; 0 sigma\_y];%measurement noise

Q = [0 0 0 0; 0 0 0 0; 0 0 0.001 0; 0 0 0 0.001];%process noise

xt = X(1);

yt = X(2);

vx = 0;

vy = 0;

for i= 1:nframes-1

X = A\*X;

P = A\*P\*A'+Q;

K = P\*H'\* pinv(H\*P\*H'+R);

Z = [mx(i+1);my(i+1)];

X = X + K\*(Z-H\*X);

P = (I-K\*H)\*P;

xt = [xt;X(1)];%position

yt = [yt;X(2)];

vx = [vx,X(3)];%velocity in x direction

vy = [vy,X(4)];%velocity in y direction

end

%

shapeInserter2 = vision.ShapeInserter('Shape','Circles','BorderColor',...

'Custom','CustomBorderColor',uint8([0 255 255]));

v2 = VideoWriter('E:/2016Spring/MV/HW5/test');

open(v2);

for k = 1:nframes

circle = int32([xt(k),yt(k),30]);

tag(:,:,:,k) = step(shapeInserter2,tag(:,:,:,k),circle);

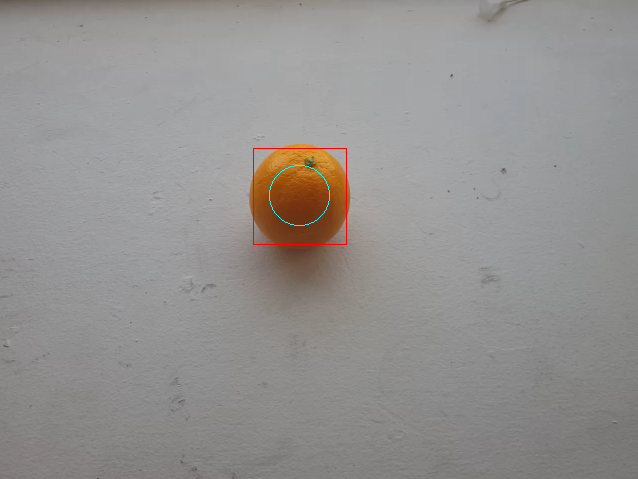
writeVideo(v2,tag(:,:,:,k));

end

close(v2);

frameRate = get(v,'FrameRate');

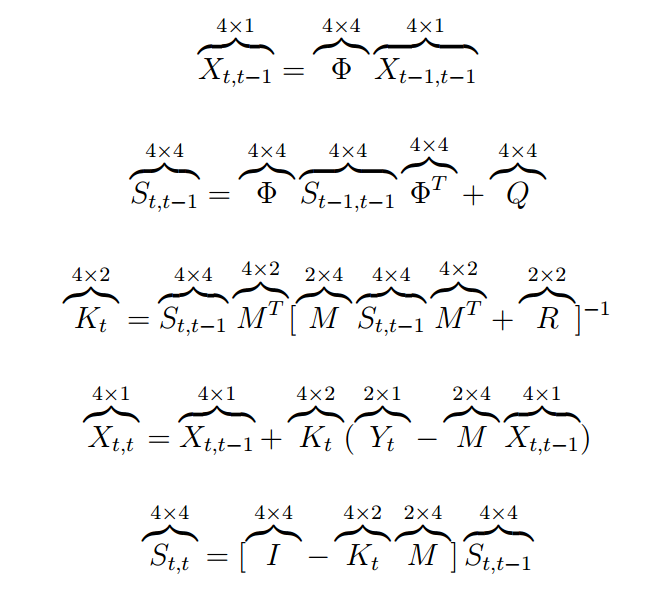
implay(tag,frameRate);



Red rectangle represent the colored base tracking, blue circle represent 2D Kalman filter.

Algorithm

2D Kalman filter



As for the lighting change, we should choose a specific range of intensity to make sure we can detect the colored object. Another way is instead of using RGB color space, we can use HSV which can separate the brightness effect.

We can simply use the ratio between velocity of x and y direction to represent its orientation.

Using 2D Kalman filter to predict its position and velocity, it helps tracking.